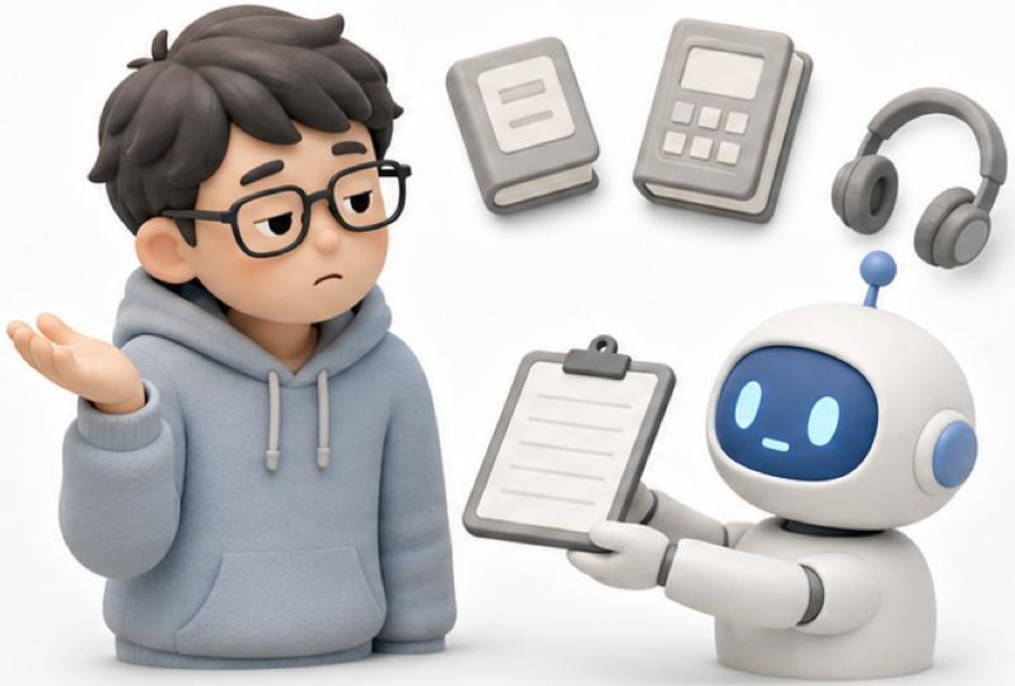
Part 3 ／ 90分

AI駆動開発の体験

まだ何もインストールしません。
ブラウザだけで”作れる”を体験します



同じAI、同じテーマ。聞き方だけを変える



素朴な質問：

『TOEICのスコアを100点あげたいです。どうすればいいですか』

単語を覚える・公式問題集を解く・毎日英語を聞く



文脈を渡した質問：『私は工学部4年の大学生で、
現在のTOEICスコアは650点（リスニング300点、リーディング350点）
です。次回の試験は1カ月後で、スコアを100点上げたいです。
平日は研究とアルバイトで1日1時間程度、休日は4時間取れます。
戦略、重点課題、週単位の勉強スケジュールを提案してください。』

自分専用の1カ月計画が返ってくる

なぜ差が出るのか ― AIはあなたを知らない



研究室の先輩は、あなたの学年も忙しさも顔も知っている



AIの手元にあるのは、あなたが打ち込んだ文字だけ

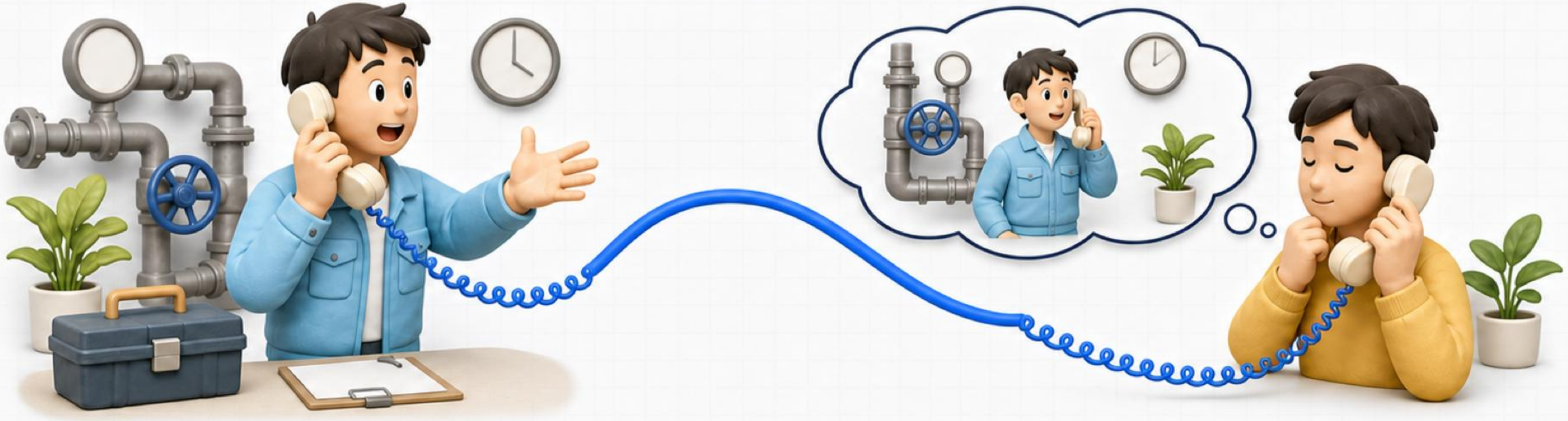
要点：　AIの性能が低いのではない。判断材料を渡していない

渡す5点

1. **前提**（自分は何者か）
2. **現状**（いまどこにいるか）
3. **制約**（使える時間・資源）
4. **目標**（何を達成したいか）
5. **出力形式**（どんな形で欲しいか）

電話メソッド

いま目の前にある光景を、電話だけで相手に伝えるとしたら？

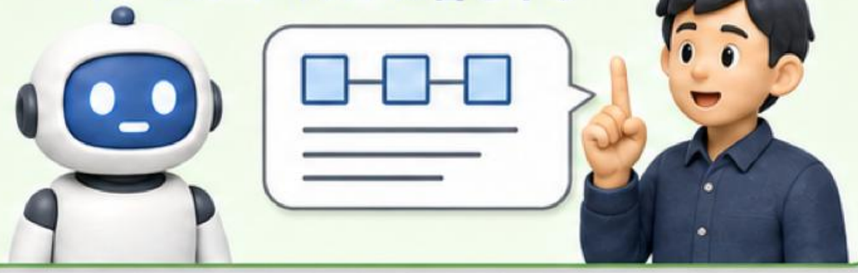


聞いた人がその場面をリアルに思い浮かべられるだけの情報量を、言葉で渡す

日本の職場・研究室は
"察してもらう"ハイコンテキスト文化



AIは察しない ―
誰が読んでも、そうしか読めないところまで
ローコンテキストに書き下す



構造化プロンプト ― 目的・制約・出力形式を分けて書く

―― 長い依頼は、機能ごとに区切る。区切れていれば、記号は何でもよい ――



見出し（### 要件）
でも、タグでもよい。
細部は気にしない

```
<目的> 定例記者会見の
要旨レポートを作成する </目的>

<指示>
- 記者の質問と所属を抽出する
- 回答を箇条書きでまとめる
</指示>

<出力形式>
## 質問のテーマ（記者名: 所属）
／ Q: 質問の要旨 ／ A: 回答の要旨
</出力形式>

<入力>
ここに対象データを貼る
</入力>
```

深津式プロンプト（#命令書 ／ #制約条件 ／ #入力文 ／ #出力文）も、この構造化の一例

プロンプトエンジニアリングから、コンテキストエンジニアリングへ

1 これまで：聞き方の工夫 — 効く言い回しを探す



良い質問者から、
良い情報提供者へ

いま：何を知らせるかの設計 — 判断材料を揃えて渡す



2

要点： モデルが賢くなるほど、言い回しの効きは薄れ、渡した材料の質が結果を決める

【ワーク】Google AI Studio で、質問を設計する



1 aistudio.google.com を開き、
Googleアカウントでログイン



2 自分のテーマで、
まず素朴に1回聞く



3 前提・現状・制約・目標・
出力形式を足して、
もう1回聞く



4 2つの回答を並べて
違いを見る



お題は自由 — 研究テーマの進め方、
就職活動、学会発表の準備、趣味でもOK

入力してよいのは公開情報だけ

【Tips】音声入力で、長い文脈を渡す

厚いコンテキストを渡す最大の障害は"打つのが面倒"



1. OSの音声入力:
   Windows は Win キー + H、
   Mac は Fn キーを2回押す
   (設定により異なる)

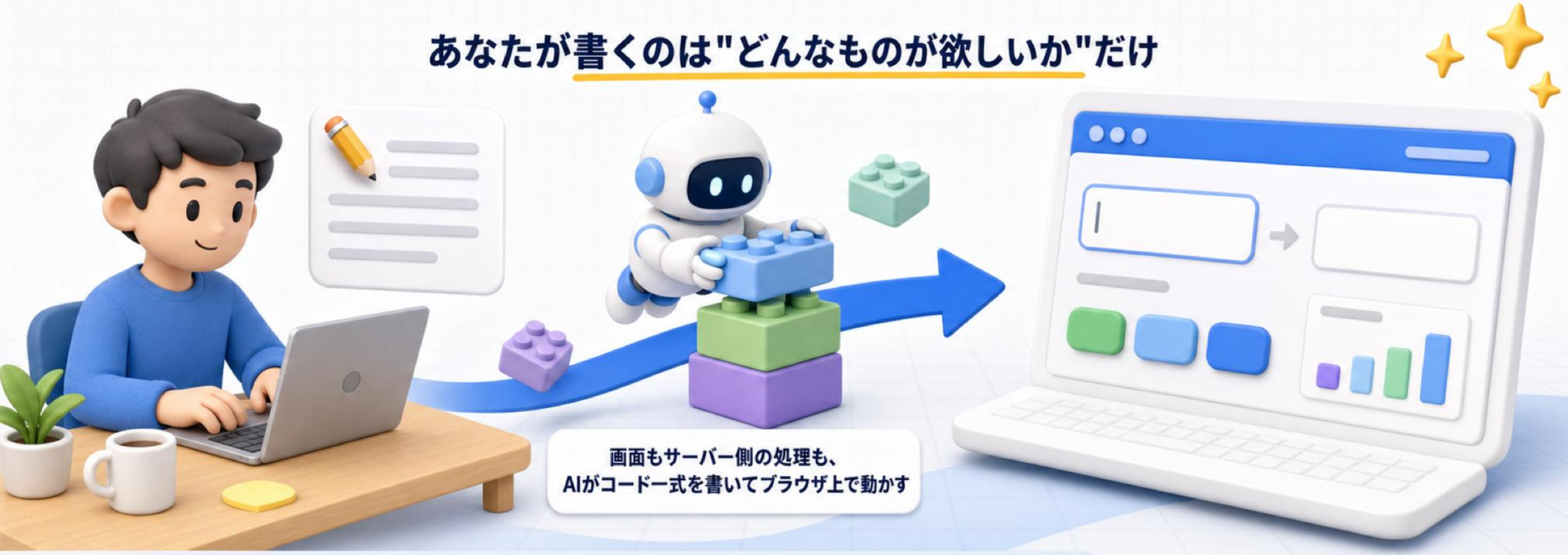
2. 話し言葉のまま、
   だらだら話してよい。
   AIは意図を汲む

3. 音声対話モードは
   優秀な助手と話す感覚。
   発言が文字で残るので
   メモにもなる

Build — 自然言語が、動くWebアプリになる

作りたいものを文章で書くと、その場で動くアプリが生成される

あなたが書くのは"どんなものが欲しいか"だけ

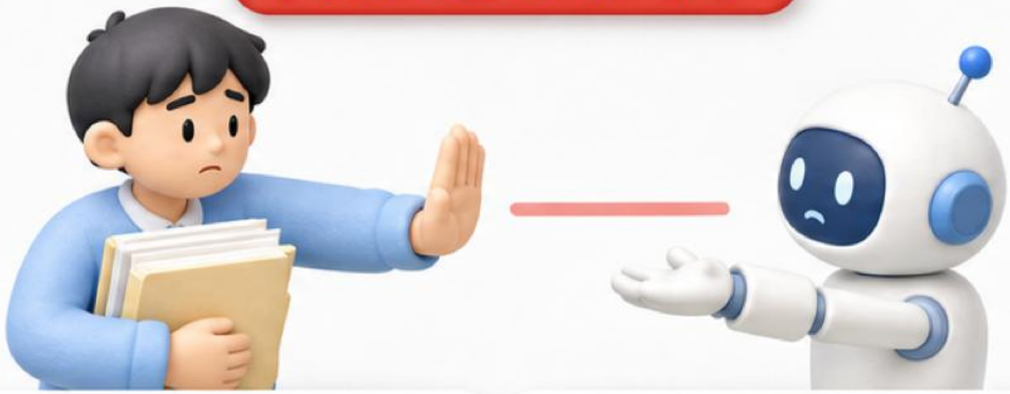


画面もサーバー側の処理も、
AIがコード一式を書いてブラウザ上で動かす

要点： 魔法ではない — さっき練習した"文脈を渡す"が、そのまま効く

無料のAIサービスに、入れてよいもの・入れてはいけないもの

入れてはいけない





個人情報（氏名・連絡先・学籍番号）



未公開の研究データ・測定値



研究室や企業の機密資料



パスワード・APIキー



個人の無料アカウントは、入力内容が学習に使われる設定の場合がある

入れてよい





公開情報



教科書レベルの一般知識



自分で作った架空データ



要点： 迷ったら入れない。迷ったこと自体が答え

【ワーク】Build でミニアプリを作る

お題例：単位換算ツール ／ ポモドーロタイマー ／ おみくじ









自由テーマでもOK ― 自分が欲しいものがあるなら、それが最良のお題

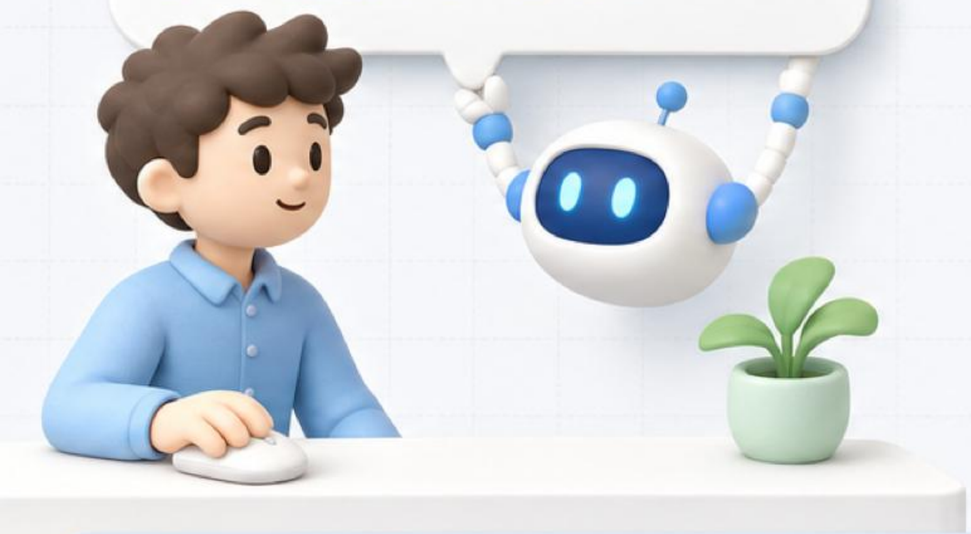
書き方の型：何をするアプリか ＋ 画面にあるもの ＋ 操作の流れ ＋ 見た目の希望

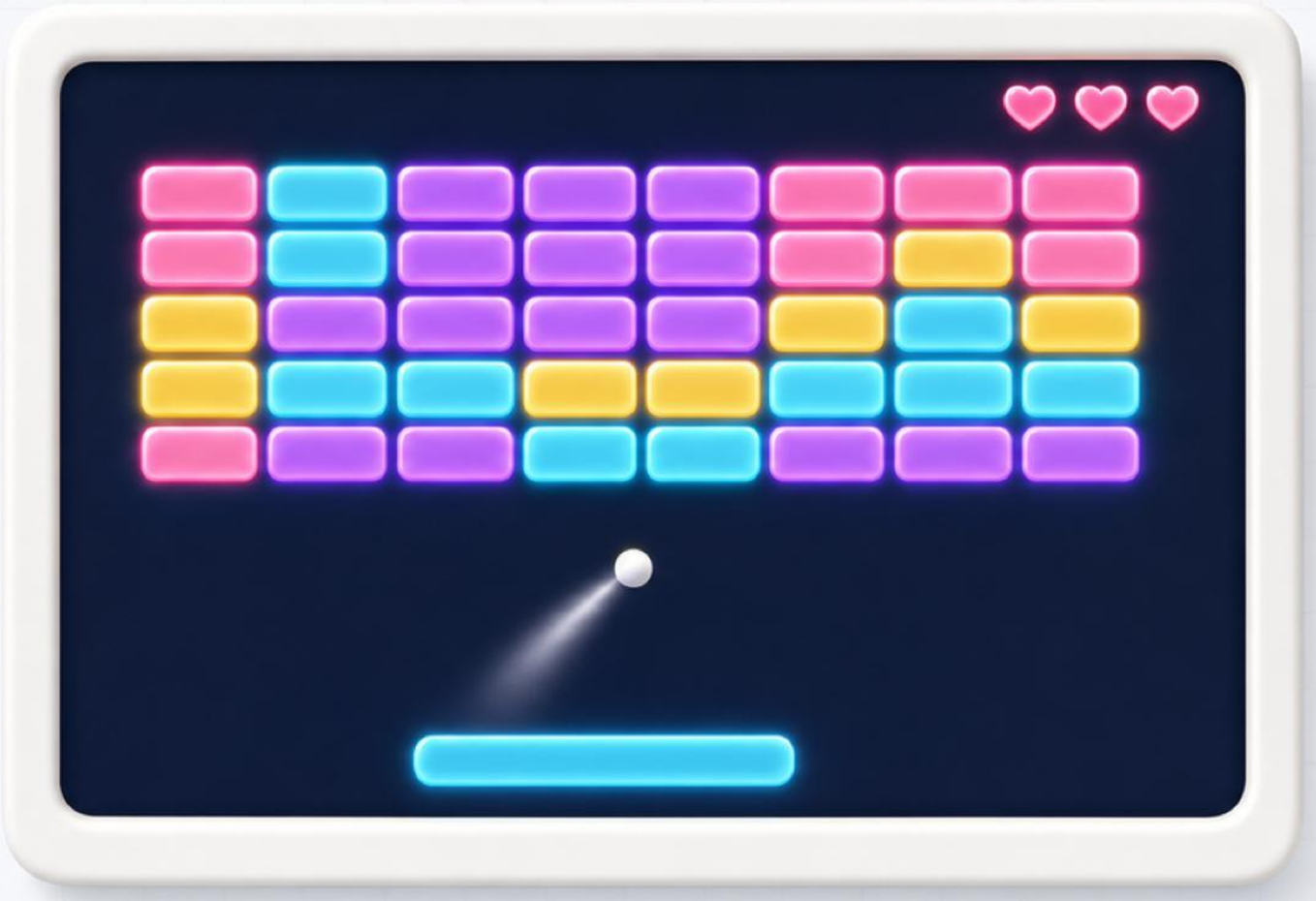
例：『長さの単位を変換するWebアプリを作ってください。m・cm・inch・feet に対応し、数値を入力すると他の単位に即時変換して表示します。スマートフォンでも押しやすい大きなボタンにしてください。配色は濃紺と水色で。』

入力するのは公開情報だけ（前のページのルール）

【事例】ゲームも作れる ── ブロック崩し

依頼文例：『ブロック崩しゲームを作ってください。マウスでパドルを操作し、ボールを跳ね返してブロックを消します。スコアと残機3を表示し、全部消したらクリア画面を出してください。配色はネオン風で。』





ゲームのルールを日本語で説明できれば、それがそのまま仕様になる

【事例】勉強道具は、自分仕様で作る ー 元素記号クイズ





依頼文例:『元素記号の4択クイズを作ってください。元素名から記号を選ぶ問題を10問出題し、最後に正答率と間違えた問題の一覧を表示してください。出題範囲を「原子番号1~20」と「全元素」から選べるようにしてください。』

暗記科目なら何でもクイズ化できる ー 専門用語・英単語・法規

要点: 探しても見つからない"欲しい仕様"は、作った方が早い

【事例】研究の道具も、その場で — CSVグラフ化ツール

依頼文例:『テキストエリアにCSVデータを貼り付けると折れ線グラフを描くWebアプリを作ってください。1列目を横軸、2列目以降を系列として扱い、系列ごとに表示・非表示を切り替えられるようにしてください。』



⚠ 貼ってよいのは公開データか、自分で作った架空データだけ(さきほどのルール)

要点: 測定データの傾向をその場で見る、使い捨ての道具が数分で手に入る

生成されたコードを覗いてみる

読めなくて大丈夫。今日つかんでほしいのは"構造がある"ことだけ



役割ごとにファイルが分かれている:









要点: 読めるようになると、直せる。直せるようになると、任せられる範囲が広がる

会話で直す ― 1回で完成させなくてよい



『スマートフォンで見るとボタンが小さいので、大きくしてください』

『ダークモードに対応させてください』

『変換結果を小数第2位までにしてください』



作る → 動かす → 気づく → 直す

うまくいかないときは、直してほしい状態を具体的に言い直す

たった今、あなたはアプリを作った

—— 環境構築ゼロ。コードは1行も書いていない ——



1時間前まで、これは"プログラムを書ける人の仕事"でした



ここが出発点です

変わったのはあなたの能力ではなく、必要な前提知識の量

なぜ、ローカル開発環境へ進むのか

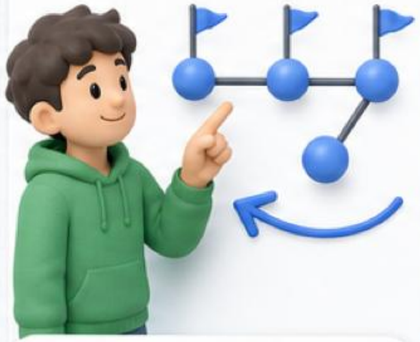


前提：Build でも保存できる。共有もできる

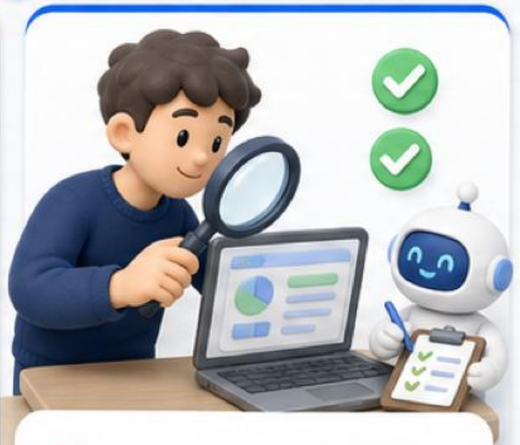
それでも次に進む理由：



1 環境と依存関係を
自分で所有する —
何が入っているかを
自分で決める



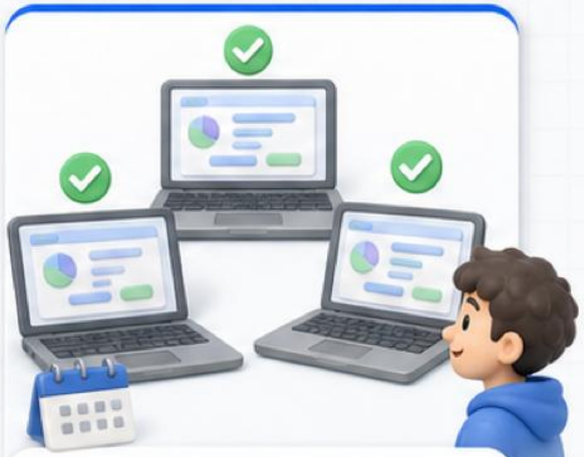
2 Git 履歴 —
変更の記録と、
いつでも戻れる
セーブポイント



3 詳細な検証 —
手元で実行し、確かめ、
テストを書かせる



4 任意のツールを使える —
好きなライブラリ・
エディタ・エージェント



5 サービスに依存しない再現性 —
他人のPCでも、来年の
自分のPCでも動く

このパートの持ち帰り



1 AIはあなたを知らない ― 前提・現状・制約・目標・出力形式を渡す



2 ローコンテキストに書き下す ― 誰が読んでもそう読むしかない文に



3 聞き方の工夫より、何を知らせるかの設計（コンテキストエンジニアリング）



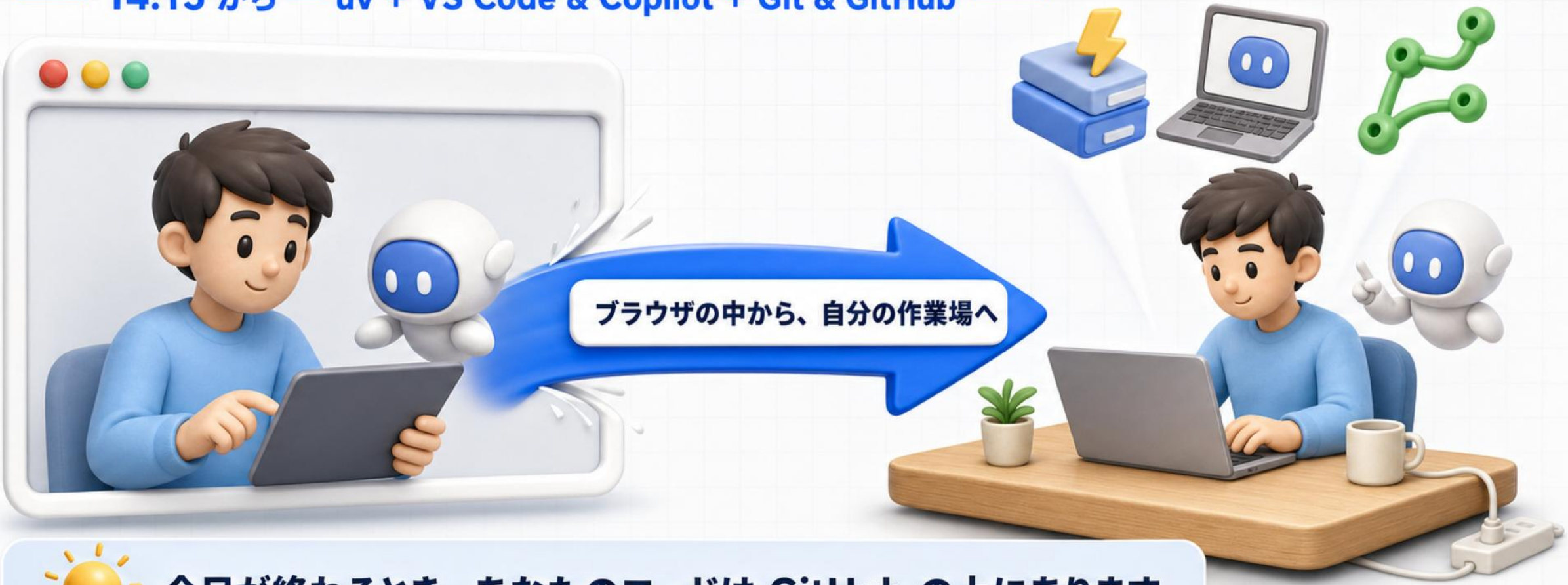
4 1回で完成させない ― 作る → 動かす → 気づく → 直す



5 検証するのは、いつも人間

午後は、自分のPCを開発環境にする

14:15 から — uv + VS Code & Copilot + Git & GitHub



今日が終わるとき、あなたのコードは GitHub の上にあります

15分休憩 — 飲み物と、PCの充電を